OIPE JC124
DEC 1 5 2003
PATENT & TRADEMARK OFFICE

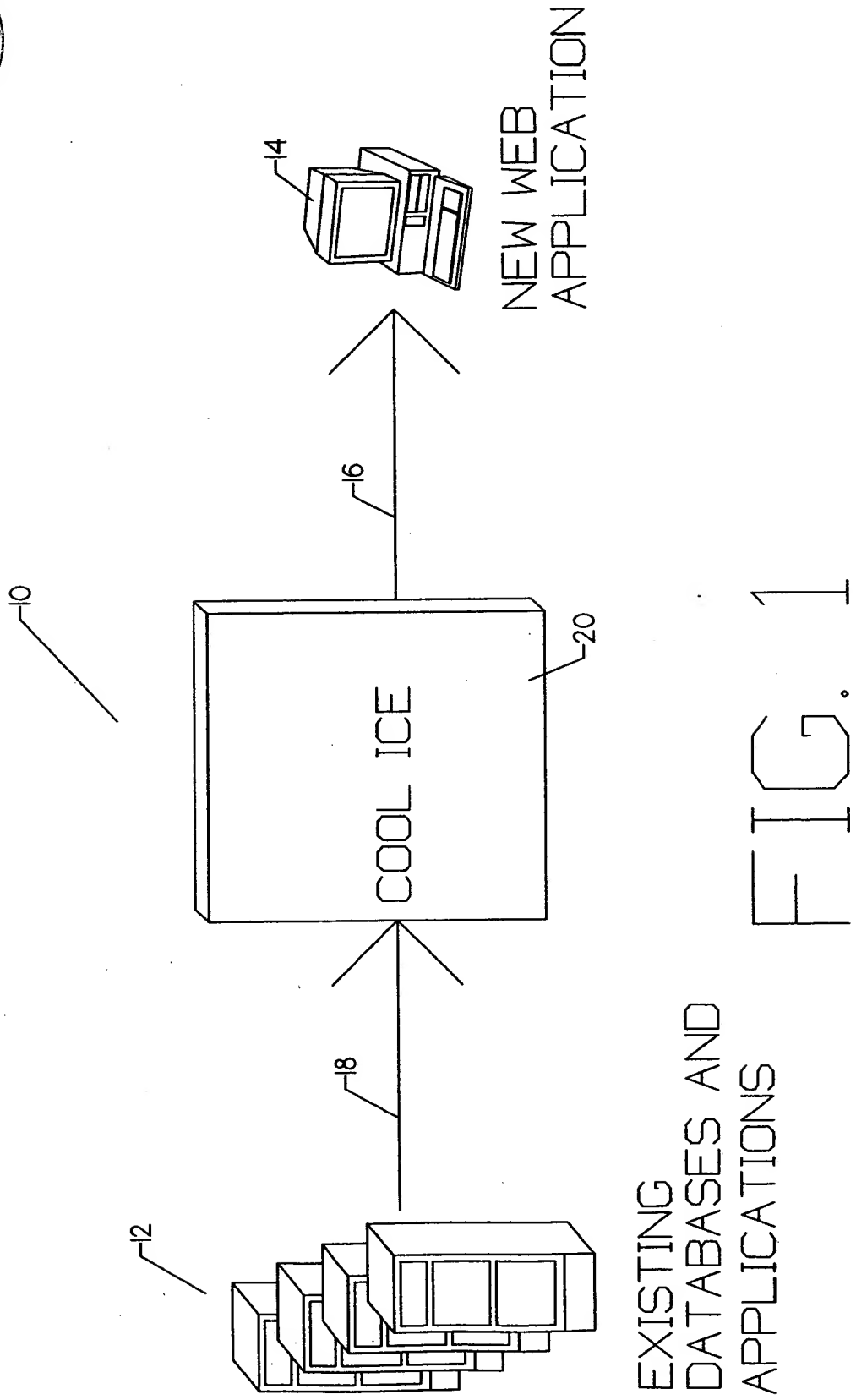

FIG. 1

OIPE
DEC 15 2003
PATENT & TRADEMARK OFFICE

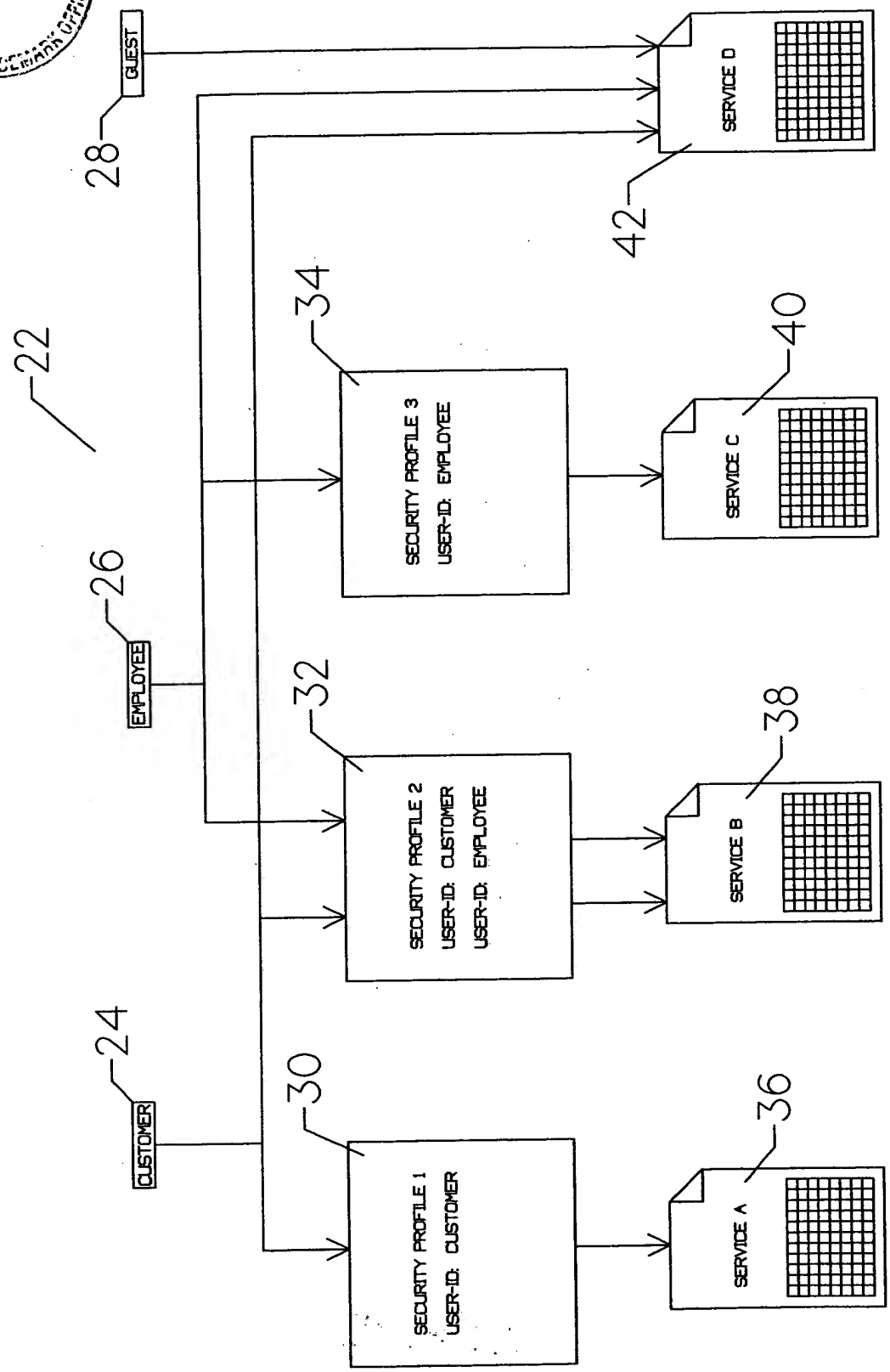

FIG. 2

OIPE JC124
DEC 15 2003
PATENT & TRADEMARK OFFICE

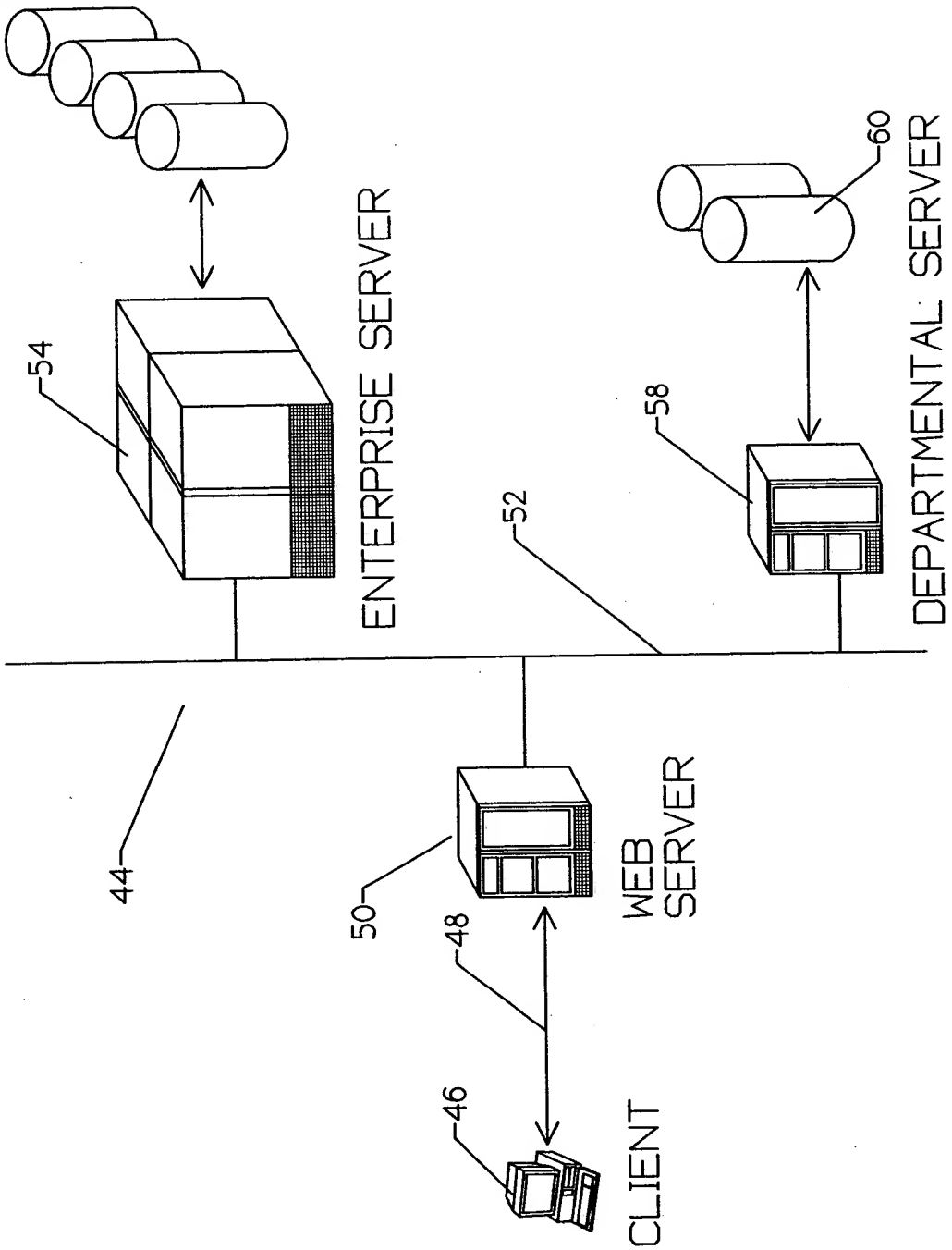

FIG. 3

OIPE
DEC 15 2003
PATENT & TRADEMARK OFFICE

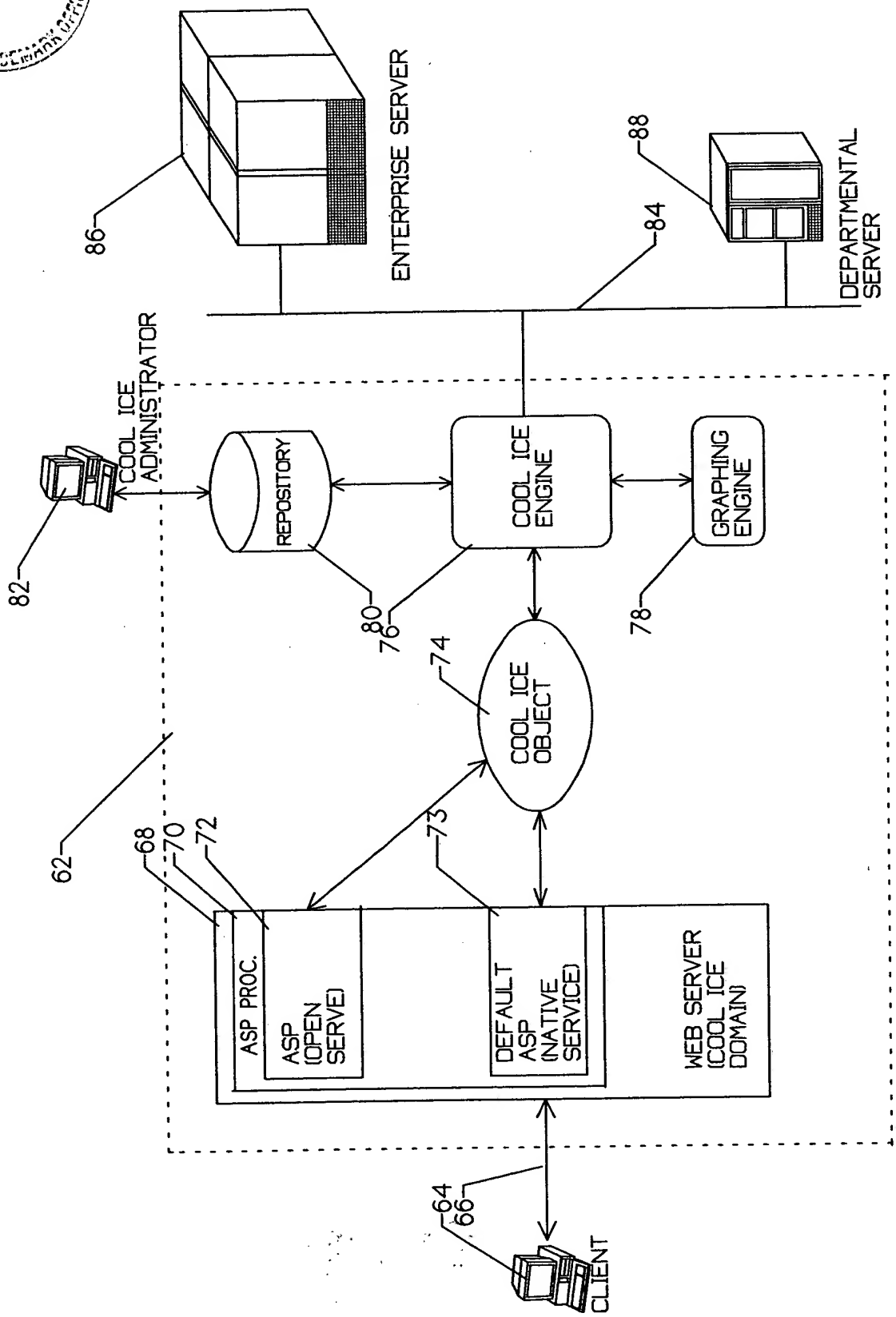

FIG. 4

OIPE
DEC 1 5 2003
PATENT & TRADEMARK OFFICE

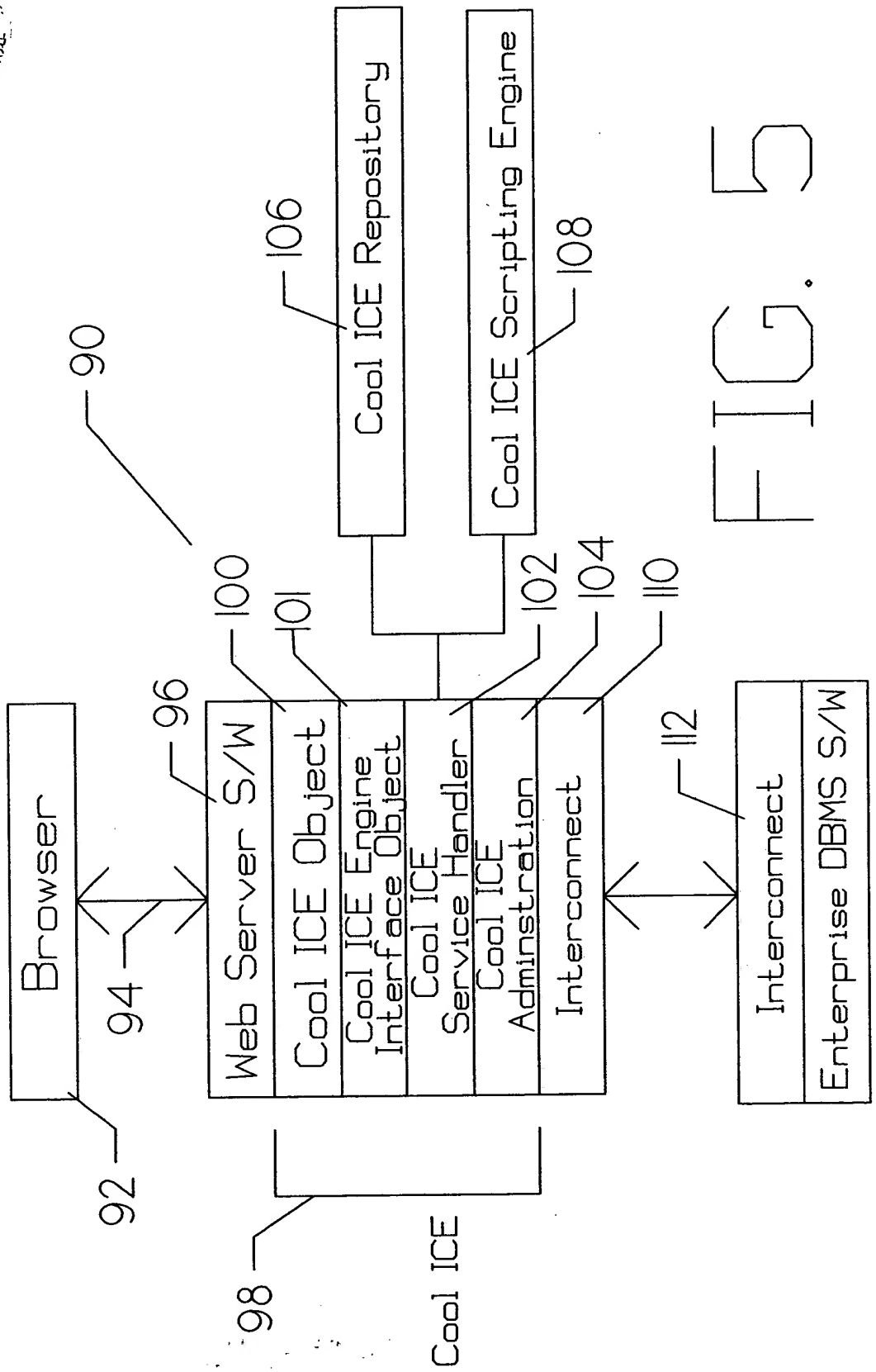

FIG. 5

OIPE
DEC 1 5 2003
PATENT & TRADEMARK OFFICE

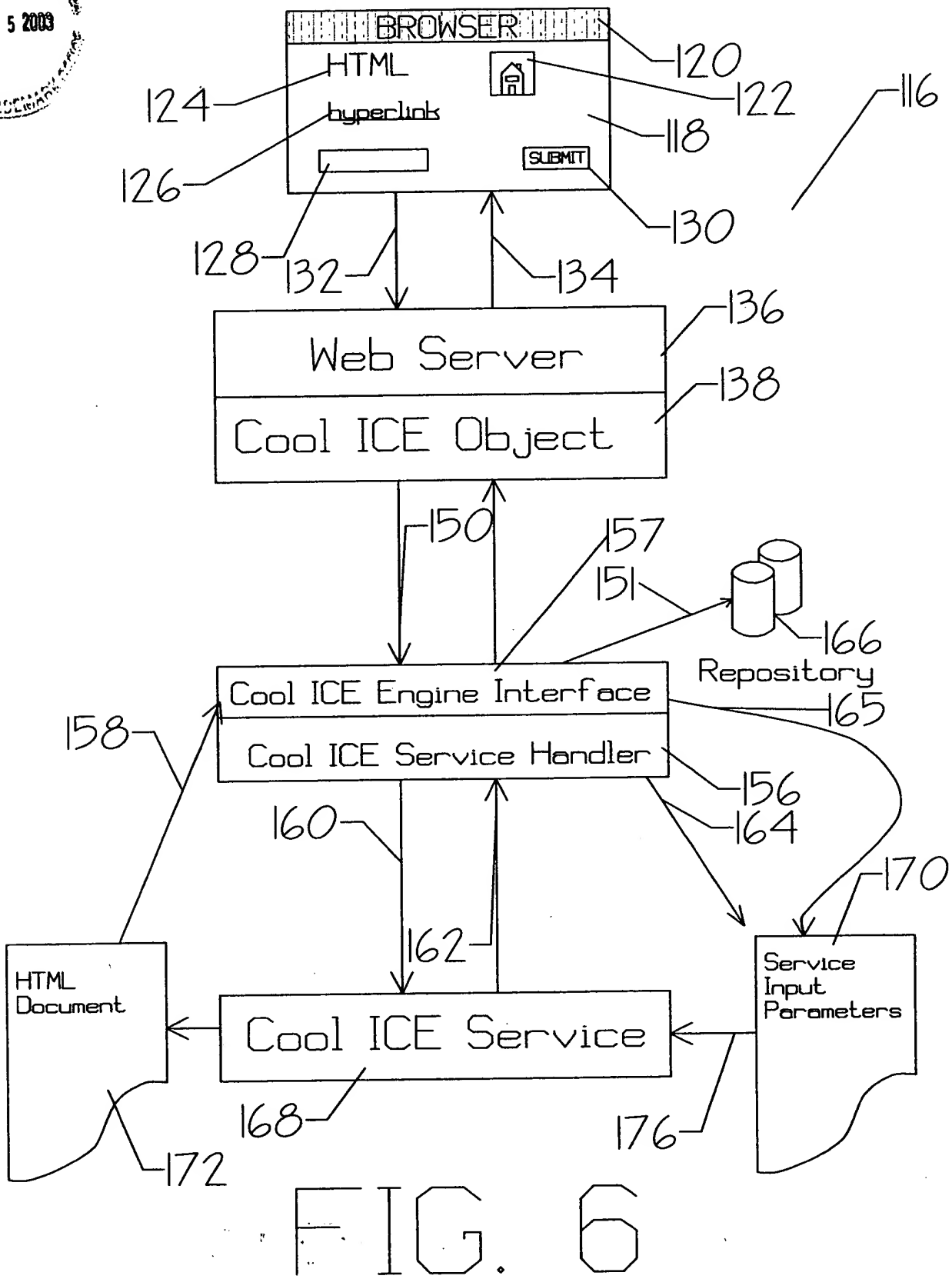

FIG. 6

OIPE
DEC. 1 5 2009
PATENT & TRADEMARK OFFICE

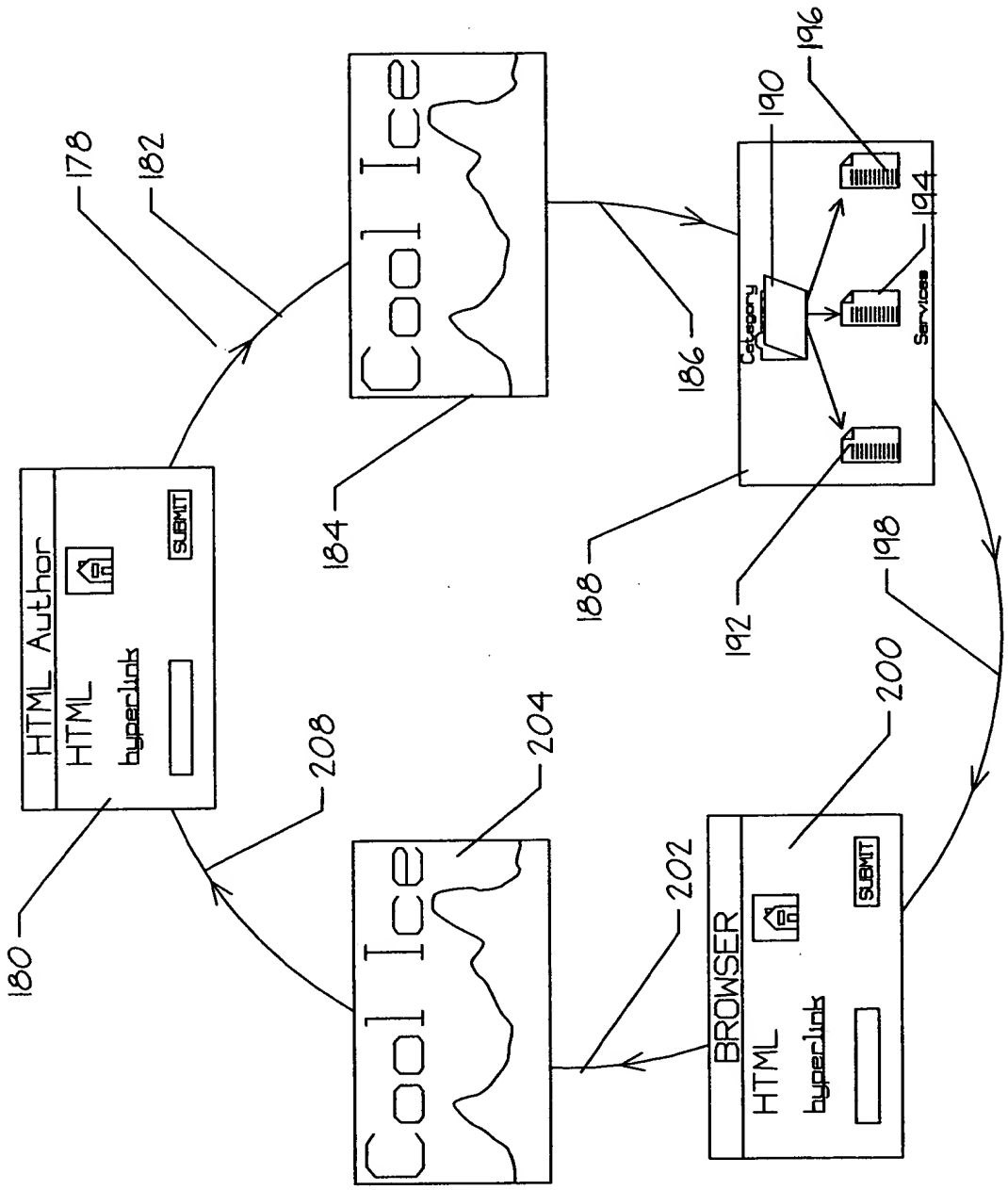

FIG. 7

OIPE
DEC 15 2003
PATENT & TRADEMARK OFFICE

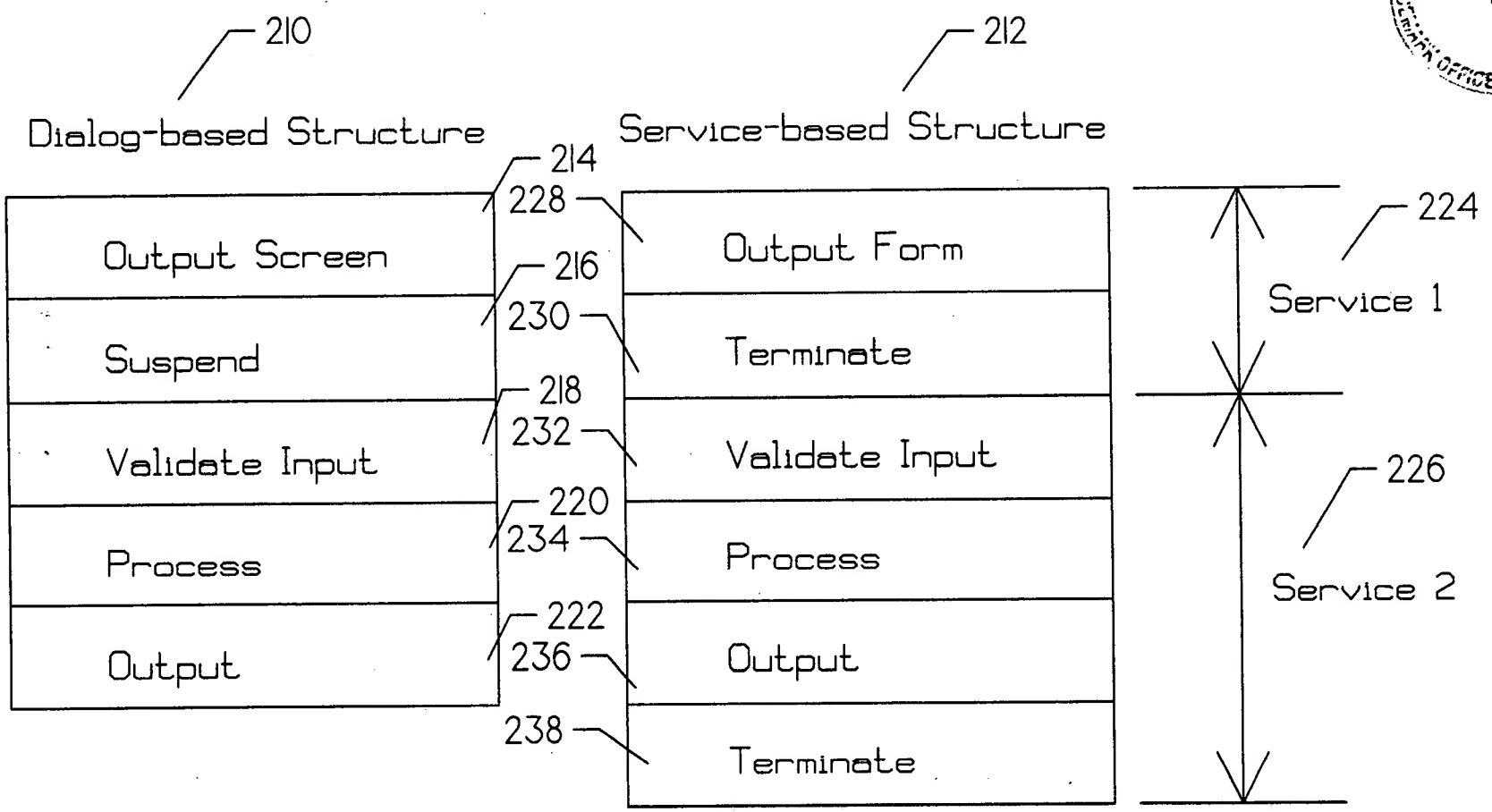

FIG. 8

REPLACEMENT SHEET

OIPE
DEC 1 5 2003
PATENT & TRADEMARK OFFICE

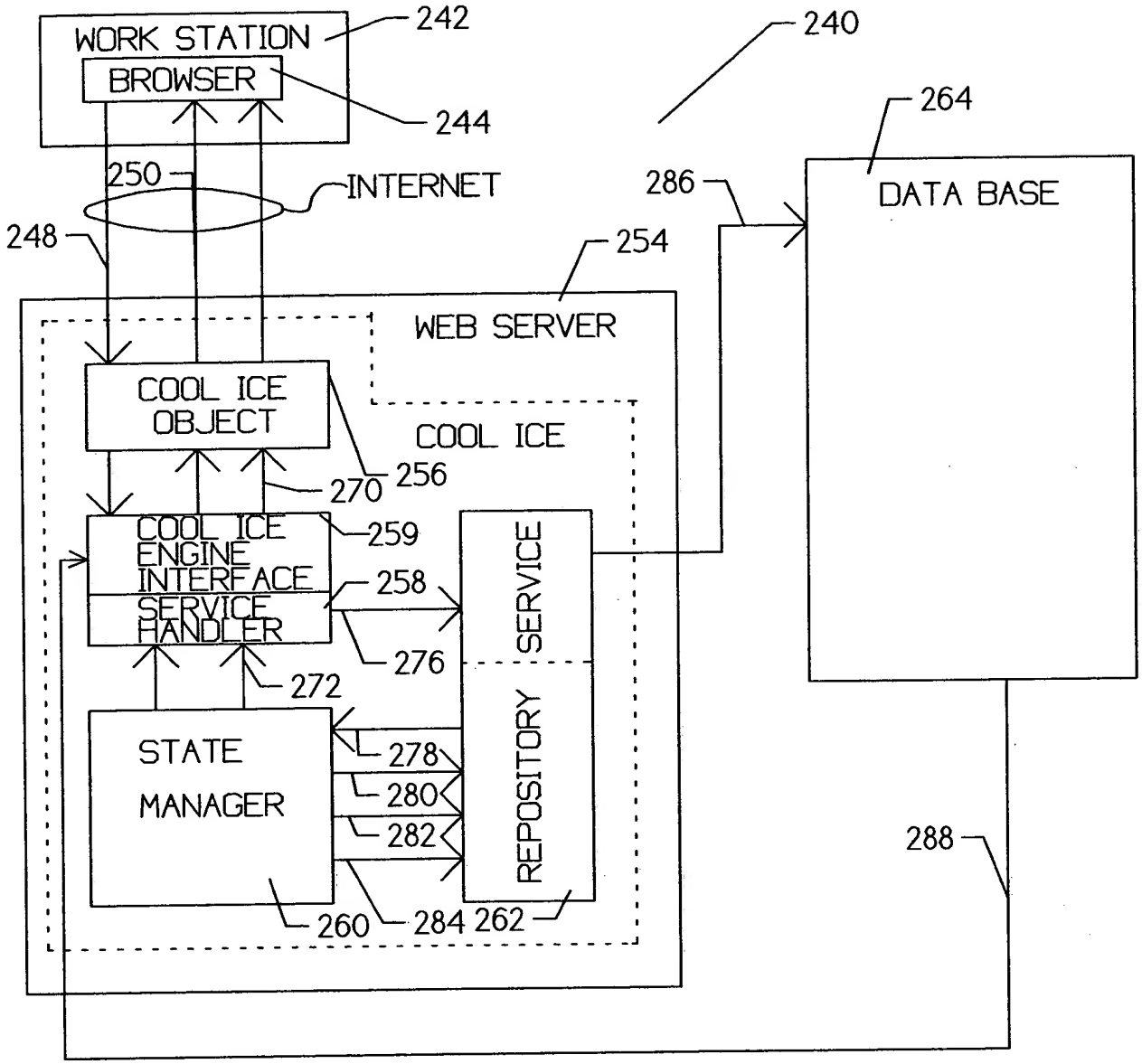

FIG. 9

OIPE
DEC 1 5 2003
PATENT & TRADEMARK OFFICE

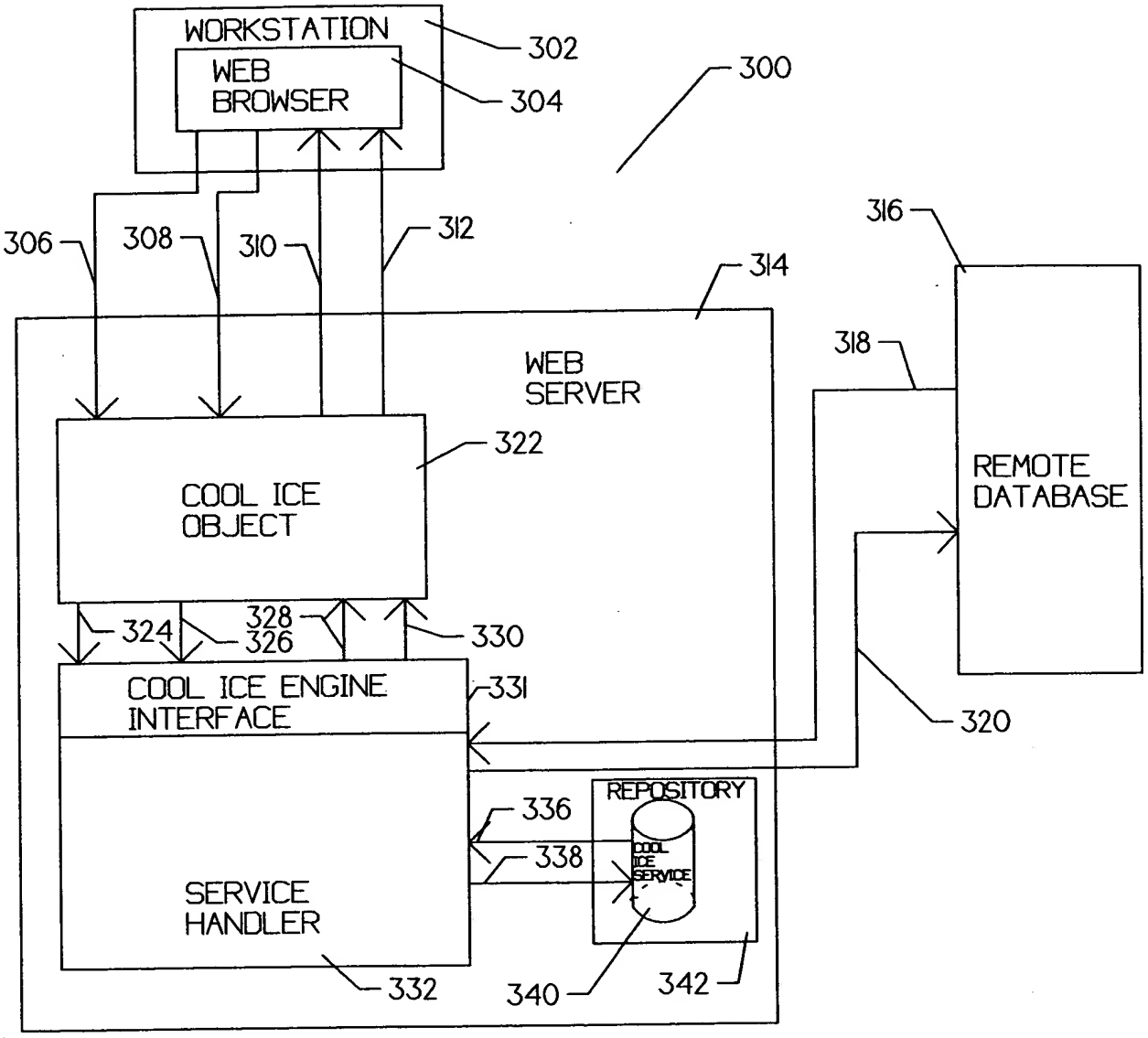

FIG. 10

OIPE
DEC 1 5 2003
PATENT & TRADEMARK OFFICE

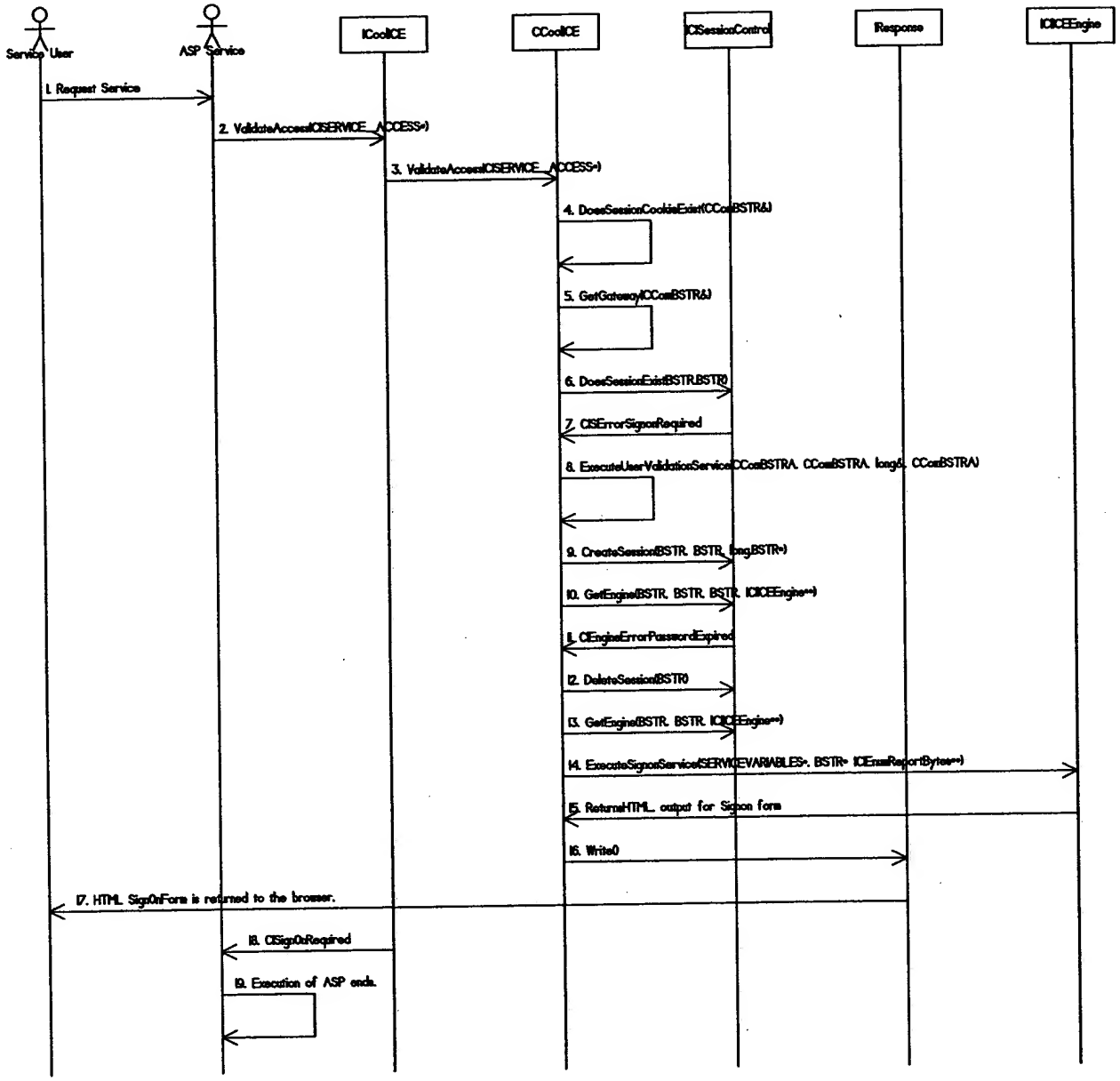

FIG. 11

BEST AVAILABLE COPY

BEST AVAILABLE COPY

PATENT & TRADEMARK OFFICE DEC 15 2003 OIPE

REPLACEMENT SHEET

MESSAGE #	DESCRIPTION
1.	The Service User will make a request for an ASP page from a browser.
2.	The ASP is executed which will create a CoolICE object and call the ValidatAccess0method. This method is intended to validate that the Service User does have access to the ASP being executed.
3.	
4.	Call the helper method DoesSessionCookieExist to determine if the CISESSIONID cookie exists in the IRequest.Cookies collection. In this sequence diagram assume S FALSE is returned.
5.	GetGateway is called to extract the Gateway Name from the ASP ServerVariables Collection of the ASP Request object. The PATH_INF0 variable will return the part of the URL after the server name but before any query string. The gateway name would be the first directory in the PATH_INF0. For example: URL Request: http://MyServer/CoolICE/abc.asp PATH_INF0:/CoolICE/abc.asp Gateway: CoolICE
6.	DoesSessionExists0 is called to determine if a the HTML SignOn form needs to be processed. The bstrSessionID parameter is set to an empty BSTR. The bstrGatewayName parameter is the value returned by GetGateway0:
7.	DoesSessionExist0 has determined that a CCISession object does not exist and a signon is required. The CISCErrorSignonRequired HRESULT from DoesSessionExists0 is described in sequence diagrams SC02 SC04. SC05. and SC07.
8.	Call ExecuteUserValidationService to process the SignOn form input fields. In this sequence diagram assume S_OK is returned which indicates that a UserID. Department. and Password are returned. Optionally. a New Password may also be returned.
9.	CreateSession0 is called to create a CCISession object. The parameters are set as follows: -bstrGatewayName is the value returned by GetGateway0 -bstrUserID is the value of the bstrUserID parameter from the call to ExecuteUserValidationService0 -bstrPassword is the value of the bstrPassWd parameter from the call to ExecuteUserValidationService0 -nDepartment is the value of the nDept parameter from the call to ExecuteUserValidationService0 -pbstrSessionID is the address of a local variable.
10.	In order to validate that the Service User has access to the ASP. a CoolICE engine is required. In addition. if the Service User does have access. then the CoolICE engine will be needed to allow the ASP to execute additional CoolICE services. Therefore, ICISessionControl::GetEngine0 is called to access an instance of a CoolICE engine that is managed by a Connection Pool. The bstrSessionID parameter is a unique identifier for the Service User. This identifier is returned by the ICISessionControl::CreateSession0 method. The bstrGatewayName parameter is the value returned by GetGateway0. The bstrNewPassword parameter is the value of he bstrNewPassword parameter returned by ExecuteUserValidationService0. In most cases. this parameter will be an empty string. except when the current password has expired. and a new passowrd was specified.
11.	The call to GetEngine has returned the HRESULT value CIEngineErrorPasswordExpired which indicates that the specified password has expired. A specid SignOn form must now be displayed which allows the user to change their password.
12.	Since an Error was returned by GetEngine. it is necessary to delete the CISession object created in step 9 above. Pass the bstrSessionID value that was returned by CreateSession.
13.	It is necessary to get a Cool ICE Engine so that the SignOn service can be executed. In this case. a blank SessionID should be specified so that the default user-id/department/password will be used to sign on to the Cool ICE engine. The bstrSessionID parameter should be a blank string so that the default user-id/department/password will be used to sign on to the Cool ICE engine. The bstrGatewayName parameter is the value returned by GetGateway0. The bstrNewPassword parameter should be a blank string since we know that a new password has not been specified.
14.	The ExecuteSignOnService0 method is called to cause the SignOn service to be executed. This service will return a HTML form which has the user-id and department prefilled. A field for the old password and new password must be filled in by the user. The input parameters 'user' and 'depn' must be passed to the SignOn service when a password has expired. The parameters are passed in the BrowserInput section of the SERVICEVARIABLES structure. The values of these parameters are the bstrUserID and nDept parameters returned by ExecuteUserValidationService0.
15.	The SignOn service returns a HTML form that prompts the user for the old password and new password. The user-id and department number fields are prefilled and read-only.
16.	The HTML output for the SignOn service is passed to the Write0 method of the ASP Response object.
17.	The ASP IResponse object will send the HTML output for the SignOn form to the browser.
18.	ValidateAccess0 returns a CISignOnRequired status back to the ASP. This will cause the ASP to terminate.
19.	Execution of the ASP is terminated due to the CISignOnRequired status being returned from ValidateAccess0.

FIG. 12

OIPE JC14
DEC 1 5 2003
PATENT & TRADEMARK OFFICE

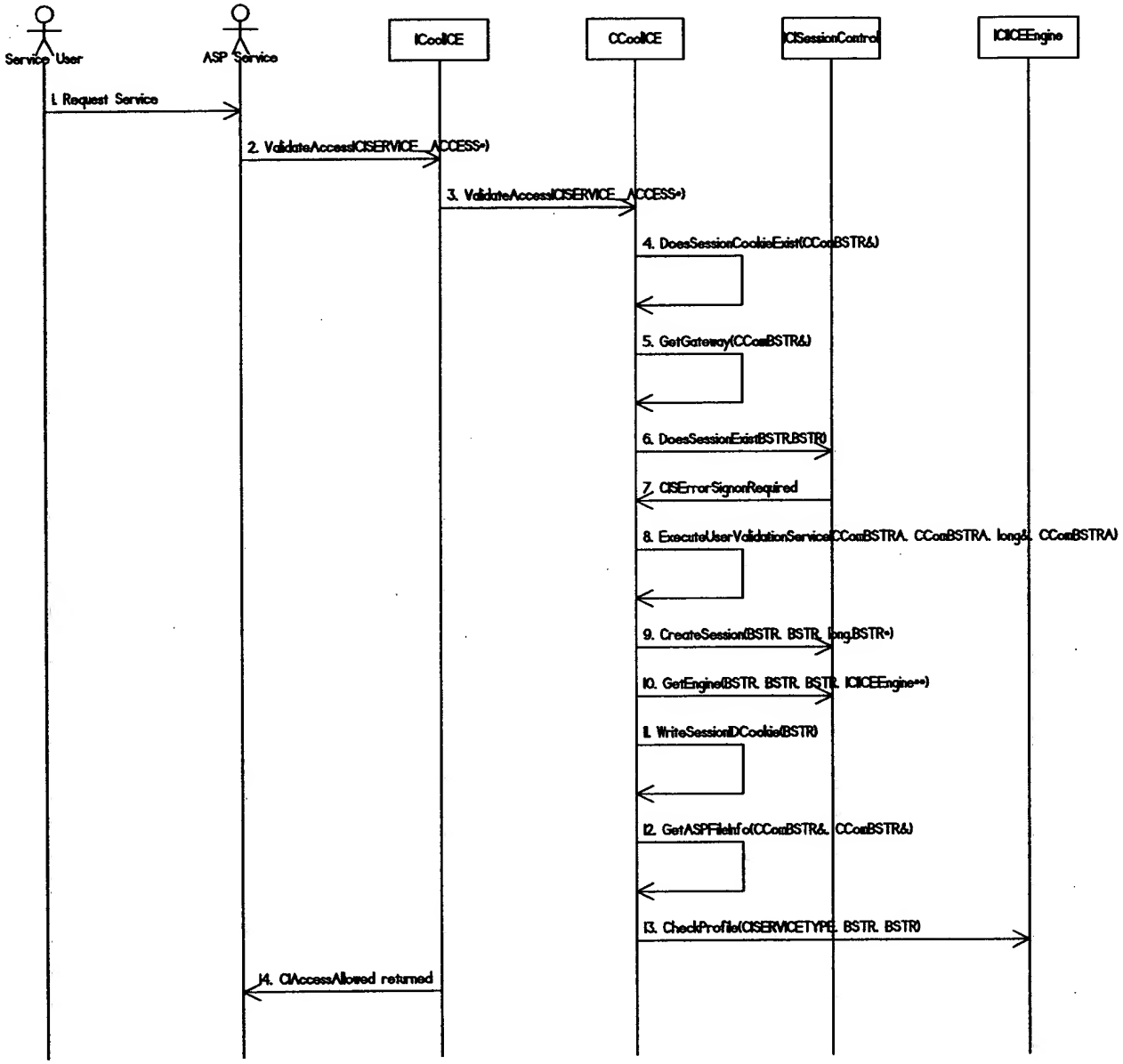

FIG. 13

BEST AVAILABLE COPY

OIPE
DEC 1 5 2003
PATENT & TRADEMARK OFFICE

MESSAGE #	DESCRIPTION
1.	The Service User will make a request for an ASP page from a browser.
2.	The ASP is executed which will create a CoolICE object and call the ValidatAccess()method. This method is intended to validate that the Service User does have access to the ASP being executed.
3.	
4.	Call the helper method DoesSessionCookieExist to determine if the CISESSIONID cookie exists in the IRequest.Cookies collection. In this sequence diagram assume S FALSE is returned.
5.	GetGateway is called to extract the Gateway Name from the ASP ServerVariables Collection of the ASP Request object. The PATH_INFO variable will return the part of the URL after the server name but before any query string. The gateway name would be the first directory in the PATH_INFO. For example: URL Request: http://MyServer/CoolICE/abc.asp PATH_INFO:/CoolICE/abc.asp Gateway: CoolICE
6.	DoesSessionExists() is called to determine if a the HTML SignOn form needs to be processed. The bstrSessionID parameter is set to an empty BSTR. The bstrGatewayName parameter is the value returned by GetGateway():
7.	DoesSessionExist() has determined that a CCISession object does not exist and a signon is required. The CISCErrorSignonRequired HRESULT from DoesSessionExists() is described in sequence diagrams SC02 SC04. SC05. and SC07.
8.	Call ExecuteUserValidationService to process the SignOn form input fields. In this sequence disgram assume S_OK is returned which indicates that a UserID. Department. and Password are returned. Optionally. a New Password is returned.

FIG. 14A

OIPE
DEC 1 5 2003
PATENT & TRADEMARK OFFICE

CreateSession() is called to create a CCISession object.

The parameters are set as follows:

-bstrGatewayName is the value returned by GetGateway()

-bstrUserID is the value of the bstrUserID parameter from the call to ExecuteUserValidationService()

-bstrPassword is the value of the bstrPassWd parameter from the call to ExecuteUserValidationService()

-nDepartment is the value of the nDept parameter from the call to ExecuteUserValidationService()

-pbstrSessionID is the address of a local variable.

10. In order to validate that the Service User has access to the ASP. a CoolICE engine is required. In addition. if the Service User does have access. then the CoolICE engine will be needed to allow the ASP to execute additional CoolICE services.

Therefore. ICISessionControl::GetEngine() is called to access an instance of a CoolICE engine that is managed by a Connection Pool.

The bstrSessionID parameter is a unique identifier for the Service User. This identifier is returned by the ICISessionControl::CreateSession() method.

The bstrGatewayName parameter is the value returned by GetGateway().

The bstrNewPassword parameter is the value of he bstrNewPassword parameter returned by ExecuteUserValidationService(). In most cases. this parameter will be an empty string. except when the current password has expired. and a new passowrd was specified.

11. Call WriteSessionIDCookie to write the SessionID value returned by ICISessionControl::CreateSession out to the browser as the CISESSIONID cookie.

12. The helper method GetASPFileInfo is called to retrieve the virtual directory alias name and the file name of the ASP.

13. The ICICEEngine::CheckProfile() method is called to verify that the user. as known to the Cool ICE engine. does have access to the ASP.

14. The ValidateAccess() method will return CIAccessAllowed status indicating that the Service User does have access. therefore. the execution of the ASP can continue.

FIG. 14B